

IN THE CLAIMS:

1 I. (Currently Amended) A method for converting a file access data structure from a first
2 endianness to a second endianness by a processor, ~~the method comprising the steps of:~~

3 determining if the file access data structure is a critical path data structure, where

4 the critical path data structure includes commonly utilized data structures;

5 in response to determining that the file access data structure is the critical path
6 data structure, performing byte swap operations using specific code functions and placing
7 a converted data structure in an output buffer to make ~~it~~ the converted data structure

8 available for further processing;

9 in response to determining that the file access data structure is not the critical path
10 data structure, calling a byte-swapping engine to perform the following:

11 a.) identifying, from a descriptor look up table, a series of actions to per-
12 form on elements of the file access data structure;

13 b.) performing the identified series of actions on the elements of the file
14 access data structure to convert the file data structure from the first endianness to
15 the second endianness; and

16 c.) placing a converted data structure in an output buffer to make ~~it~~ the
17 converted data structure available for further processing.

2.-19. (Cancelled)

20. (Currently Amended) A method for converting a data structure by a processor, comprising:

determining if the data structure is a critical path data structure, where the critical path data structure includes commonly utilized data structures;

in response to determining that the data structure is the critical path data structure, performing byte swap operations using specific code functions and placing a converted data structure in an output buffer to make ~~it the converted data structure~~ available for further processing and placing a converted data structure in an output buffer to make it available for further processing;

in response to determining that the data structure is not the critical path data structure, calling a byte-swapping engine and performing the following:

a.) providing a file access data structure as input to the byte-swapping engine;

b.) providing a descriptor look up table to the byte-swapping engine;

c.) identifying, from the descriptor look up table, a series of actions to perform on elements of the file access data structure in order to swap bytes of the file access data structure from a first endianness to a second endianness;

d.) performing the identified series of actions on the elements of the file access data structure to convert the file access data structure; and

e.) placing a converted data structure in an output buffer to make ~~it the converted data structure~~ available for further processing.

1 21. (Previously Presented) The method as in claim 20, further comprising:
2 using as the file access data structure a file having Direct Access File System
3 (DAFS) protocol.

1 22. (Cancelled)

1 23. (Currently Amended) A method for converting a data structure by a processor, com-
2 prising:

3 determining if the data structure is a critical path data structure, where the critical
4 path data structure includes commonly utilized data structures;

5 in response to determining that the data structure is a critical path data structure,
6 performing byte swap operations using specific code functions and placing a converted
7 data structure in an output buffer to make ~~it~~ the converted data structure available for fur-
8 ther processing;

9 in response to determining that the data structure is not the critical path data struc-
10 ture, calling a byte-swapping engine and performing the following:

11 a.) providing a file access data structure as input to the byte-swapping en-
12 gine;

13 b.) providing a descriptor look up table to the byte-swapping engine;

14 c.) identifying, from the descriptor look up table, a series of actions to per-
15 form on elements of the file access data structure in order to swap bytes of the file
16 access; and

17 d.) performing the identified series of actions on the elements of the data
18 structure header to convert the file access data structure; and

19 e.) placing a converted data structure in an output buffer to make ~~it~~ the
20 converted data structure available for further processing.

1 24. (Previously Presented) The method as in claim 20, further comprising:
2 swapping bytes of the data structure as needed, in response to swapping bytes of
3 the file access data structure.

1 25. (Previously Presented) The method as in claim 20, further comprising:
2 determining if an element entry of the descriptor look up table is nested;
3 branching to the nested entry;
4 identifying, from the descriptor look up table, a nested series of actions to perform
5 on elements of the nested entry in order to swap bytes of the entry from a first endianness
6 to a second endianness, where the nested series of actions includes linking and convert-
7 ing.

1 26. – 27. (Cancelled)

1 28. (Currently Amended) A computer to convert a data structure by a processor, com-
2 prising:
3 means for determining if the data structure is a critical path data structure, where
4 the critical path data structure includes commonly utilized data structures; ;
5 means for performing byte swap operations using specific code functions in re-
6 sponse to determining that the data structure is the critical path data structure, and placing

a converted data structure in an output buffer to make ~~it~~ the converted data structure available for further processing;

means for calling a byte-swapping engine performing the following in response to determining that the data structure is not the critical path data structure:

a.) means for providing a file access data structure as input to the byte-swapping engine;

b.) means for providing a descriptor look up table to the byte-swapping engine; and

c.) means for identifying, from the descriptor look up table, a series of actions to perform on elements of the file access data structure in order to swap bytes of the file access data structure from a first endianness to a second endianness; and

d.) means for placing a converted data structure in an output buffer to make ~~it~~ the converted data structure available for further processing.

29. (Currently Amended) A computer to convert a data structure by a processor, comprising:

means for determining if the data structure is a critical path data structure, where the critical path data structure includes commonly utilized data structures;

means for calling a byte swapping engine and performing the following in response to determining that the data structure is not the critical path data structure:

7 means for providing a file access data structure as input to the byte-swapping en-
8 gine;

9 means for providing a descriptor look up table to the byte-swapping engine;

10 means for identifying, from the descriptor look up table, a series of actions to per-
11 form on elements of the data structure header in order to swap bytes of the file access
12 data structure from a first endianness to a second endianness; and

13 means for placing a converted data structure in an output buffer to make it ~~it~~the
14 converted data structure available for further processing.

1 30. (Previously Presented) The computer as in claim 29, further comprising:

2 means for swapping bytes of the data structure as needed, in response to swapping
3 bytes of the file access data structure.

1 31. (Previously Presented) The computer as in claim 29, further comprising:

2 means for determining if an element entry of the descriptor look up table is
3 nested;

4 means for branching to the nested entry; and

5 means for identifying, from the descriptor look up table, a nested series of actions
6 to perform on elements of the nested entry in order to swap bytes of the entry from a first
7 endianness to a second endianness, where the nested series of actions includes converting
8 and linking.

1 32. (Currently Amended) A computer readable ~~media~~medium, comprising:
2 a processor;
3 said computer readable ~~media~~medium containing instructions for execution on a
4 the processor for the practice of a method for converting a data structure ~~by a processor~~,
5 the method having the steps of,
6 determining if the data structure is a critical path data structure, where the critical
7 path data structure includes commonly utilized data structures;
8 in response to determining that the data structure is the critical path data structure,
9 performing byte swap operations using specific code functions and placing a converted
10 data structure in an output buffer to make ~~it~~the converted data structure available for fur-
11 ther processing;
12 in response to determining that the data structure is not the critical path data struc-
13 ture, calling a byte-swapping engine and performing the following:
14 a.) providing a file access data structure as input to the byte-swapping en-
15 gine;
16 b.) providing a descriptor look up table to the byte-swapping engine;
17 c.) identifying, from the descriptor look up table, a series of actions to per-
18 form on elements of the file access data structure in order to swap bytes of the file
19 access data structure from a first endianness to a second endianness;
20 d.) performing the identified series of actions on the elements of the file
21 access data structure to convert the file access data structure; and

22 e.) placing a converted data structure in an output buffer to make ~~it~~the
23 converted data structure available for further processing.

1 33.-36. (Cancelled)

1 37. (Currently Amended) A method for converting a first data structure to a second data
2 structure by a processor, the method comprising the steps of:

3 determining if the first data structure is a critical path data structure, where the
4 critical path data structure includes commonly utilized data structures;

5 in response to determining that the first data structure is not the critical path data
6 structure, calling a byte-swapping engine;

7 using a descriptor lookup table, by the byte-swapping engine, to provide actions
8 to be performed on each element of the first data structure;

9 stepping through the descriptor table, by the byte-swapping engine, and process-
10 ing each element of the first data structure according to the element's size and action to
11 convert the first data structure having an element with a first endianness into the second
12 data structure having an element with a second endianness; and

13 placing the second data structure in an output buffer to make ~~it~~the second data
14 structure available for further processing.

1 38. (Previously Presented) The method of claim 37, further comprising:

2 using a byte as the data structure.

39 - 47 (Cancelled)

1 48. (Currently Amended) The method of claim 37, further comprising:
2 determining if the first data structure is a critical path data structure, and if ~~it~~first
3 data structure is a critical path data structure then converting an element of the first data
4 structure from a first endianness to a second endianness using a set of specific code func-
5 tions.

1 49. (Previously Presented) The method of claim 48, further comprising:
2 determining that the critical data path structure is a direct access file system
3 (DAFS) header data structure.

1 50. (Previously Presented) The method of claim 48, further comprising:
2 designing the specific code functions to rapidly convert any elements of the data
3 structure to the second endianness without using a byte swapping engine

1 51. (Currently Amended) A computer to convert a file access data structure from a first
2 endianness to a second endianness, comprising:
3 an operating system executing on the computer to determine if the file access data
4 structure is a critical path data structure, where the critical path data structure includes
5 commonly utilized data structures;

6 the computer performing byte swap operations using specific code functions in
7 response to determining that the file access data structure is the critical path data struc-
8 ture, and holding a converted data structure in an output buffer to make ~~it~~ the converted
9 data structure available for further processing;

10 the computer calling a byte-swapping engine to perform the following in response
11 to determining that the file access data structure is not the critical path data structure:

12 a.) the byte-swapping engine executing a process to identify, from a de-
13 scriptor look up table, a series of actions to perform on elements of the file access
14 data structure; and

15 b.) the byte-swapping engine to perform the identified series of actions on
16 the elements of the file access data structure to convert the file data structure from
17 the first endianness to the second endianness; and

18 c.) a memory having an output buffer to hold a converted data structure in
19 the output buffer to make ~~it~~ the converted data structure available for further
20 processing.

1 52. (Previously Presented) The computer as in claim 51, further comprising:

2 an operating system executing on the computer to use as the file access data struc-
3 ture a file having Direct Access File System (DAFS) protocol.

1 53. (Previously Presented) The computer as in claim 51, further comprising:

2 the critical path data structures are small data structures.

1 54. (Previously Presented) The computer of claim 51, further comprising:
2 the operating system to determine that the critical data path structure is a direct
3 access file system (DAFS) header data structure.

1 55. (Previously Presented) The computer of claim 51, further comprising:
2 the operating system to determine to execute the specific code functions to rapidly
3 convert any elements of the data structure to the second endianness without using a byte
4 swapping engine.

1 56. (Previously Presented) The computer as in claim 51, further comprising:
2 the operating system executing on the computer to swap bytes of a file as needed,
3 in response to swapping bytes of the file access data structure.

1 57. (Previously Presented) The computer as in claim 51, further comprising:
2 the operating system executing on the computer to determine if an element entry
3 of the descriptor look up table is nested;
4 the operating system to branch to the nested entry;
5 the operating system to identify, from the descriptor look up table, a nested series
6 of actions to perform on elements of the nested entry in order to swap bytes of the entry
7 from a first endianness to a second endianness, where the nested series of actions in-
8 cludes linking and converting.

- 1 58. (Previously Presented) the method of claim 1, further comprising:
- 2 the critical path data structures are small data structures.